A picture containing tool

Description automatically generatedSchaep Simon

Simulating massive amounts of AI agents in a battle simulator.

Supervisor: Vanden Abeele Alex

Coach: Verspecht Marijn

Graduation Work 2023-2024

Digital Arts and Entertainment

Howest.be

A close up of a card

Description automatically generated

Contents

[Abstract & Key words 2](#_Toc153482830)

[Preface 3](#_Toc153482831)

[List of Figures 4](#_Toc153482832)

[Introduction 5](#_Toc153482833)

[Literature Study / Theoretical Framework 6](#_Toc153482834)

[Research 7](#_Toc153482835)

[1. Topic 1 7](#_Toc153482836)

[1.1. Subtopic 1 7](#_Toc153482837)

[1.1.1. Subtopic 7](#_Toc153482838)

[2. Topic 2 8](#_Toc153482839)

[1.2. Subtopic 1 8](#_Toc153482840)

[1.2.1. Subtopic 8](#_Toc153482841)

[case study 10](#_Toc153482842)

[1. introduction 10](#_Toc153482843)

[2. Modelling 10](#_Toc153482844)

[2.1. Blockout 11](#_Toc153482845)

[2.2. Zbrush 12](#_Toc153482846)

[3. Texturing 12](#_Toc153482847)

[4. Shading 12](#_Toc153482848)

[5. Lighting 13](#_Toc153482849)

[Discussion 14](#_Toc153482850)

[Conclusion 15](#_Toc153482851)

[Future work 16](#_Toc153482852)

[Critical Reflection 17](#_Toc153482853)

[References 18](#_Toc153482854)

[Acknowledgements 19](#_Toc153482855)

[Appendices 20](#_Toc153482856)

# Abstract & Key words

**An abstract explains the outline of the paper concisely (the methods, results, etc.). Maximum length of 250 words, preferably both in English and Dutch.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis. Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

# Preface

***A preface is a statement of the author's reasons for undertaking the work and may include personal comments that are not directly relevant to other sections of the thesis or dissertation.* No word count limit.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis. Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

Always been amazed by big battles in games, don’t see it very often

UEBS was big inspiration

The scope is more limited than I would have wanted, more advanced/time-consuming ways will be discussed but not applied, as they could be their own research paper.

This paper is meant for people like me 2 years ago, with basic knowledge of C++ programming and game development, with an interest in big battles in games.

# List of Figures

**The list of figures lists the figures in the order in which they appear throughout the thesis. They may be numbered sequentially, or be subdivided following the chapters in which they appear.**

Figure 1: A picture showing something

Figure 2: A graph showing another thing

Figure 3.1: A tabel showing yet another thing, that appears in chapter 3.

Code snippets are images, you can find all the code in the Github repository.

# Introduction

**In the introduction, you write the background of your topic and discuss the observation that spurred you on to do this research project. Explain the purpose of the paper and present your research question(s) and the hypothesis at the end of this section. This section is typically a couple of pages long.**

Wanted to make a game with big battles before, but very quickly encountered heavy performance issues, not even allowing a few hundred units

This research focuses on how you would go about making big battles in a game, the best test would be to do it in a battle simulator

But the goal is that the findings can be applied to other games that have big battles too

This research assumes x number of armies facing each other, tries to not limit agent behavior, and allow for flexibility in their actions, also assumes a standard quality for visuals, assets will be used from the Unreal Engine marketplace

This research aims to provide an overview of what you can do to improve performance of big battles, and provide a starting point for delving deeper in the various topics.

# Literature Study / Theoretical Framework

**In the literature review, you present the secondary research you have conducted. You detail the background of your topics and write about the concepts that are relevant to the study. Assume that not every reader has the same skillset or -level as you do! This section typically requires a substantial amount of references and can be a lengthy section that requires a considerable amount of pages.**

The optimization of the simulation of large amounts of agents in a video game is a complex topic since there are many factors relating to performance. There are many different ways that you can optimize depending on the needs of your game. Some are more complicated and difficult to implement than others.

We will do our best to explain to most commonly applied methods and provide sources if you want to delve deeper into a topic.

## Data oriented programming

Data oriented programming (sometimes called Data oriented design) is a programming approach where data is separated from code, with the main goal of making more efficient use of the cpu cache, by aligning data to avoid cache misses.

In a game engine, data oriented programming is usually applied by using ECS (Entity Component System).  
This is an approach where there are entities, represented by a unique ID, that are associated with components.  
Components mark an entity to have certain functionality, and hold the data needed for that functionality.  
Systems operate on all entities that have certain components, and manipulate data accordingly.

An example of this is a movement system, which evaluates all entities that have a position and movement component. The position component just holds a position, and the movement component holds a movespeed and direction. The system then goes over all position components and updates their position based on the direction and movespeed of the movement components with the same entity ID.

The Wikipedia article on ECS has a more in-depth explanation of what it is: [https://en.wikipedia.org/wiki/Entity\_component\_system](https://en.wikipedia.org/wiki/Entity_component_system%20)

## Unreal Engine Mass

Unreal Engine has implemented their own ECS system, called Mass, with the introduction of Unreal Engine 5.0. This system uses the same concepts as ECS, but has named things a bit different. Entities are still called entities, components are called fragments, and systems are called processors.

There are some other concepts that they added, like traits, which are used to define what fragments an entity has. For example a movement trait might add a transform fragment and a movement fragment. They allow for easier configuration of entities.

Another concept they added are archetypes, which group entities with the same fragments together which are then evaluated in processors as separate chunks.

They also added tags, that are fragments without data, used to identify entities easier. An entity might have the “dead” tag to indicate it is dead. The movement processor will then not operate on archetypes with the “dead” tag.

Mass also includes lots of in-built fragments and tags that can help you set up basic functionality very quickly. It includes things like steering and avoidance, and also visualization. It is also made to work well with zonegraphs, which can be used to direct crowds. This was used extensively in the City Sample demo project, released with Unreal Engine 5.0.

The in-built visualization for Mass is quite complex and will work well enough for most projects. There are two different representation methods: instanced static meshes, and actors.  
Using instanced static meshes is the most optimized as it will create an instance of a specified static mesh at the location of every entity.  
Using actors is heavier on performance since there will be an actor spawned for every entity and there is an extra step in communicating from Mass to the actor. However, actors allow for much more options in how you can render. The most obvious way of using an actor is to add a skeletal mesh to the actor to allow bone animations to be played. But you could also add logic to the actor to for example change material when you are hit.

You can even take it a step further and put gameplay logic on the actor and communicate back to Mass to inform the entity of any changes that have to be made.

The official documentation provides a good basic overview of Mass: <https://docs.unrealengine.com/5.0/en-US/overview-of-mass-entity-in-unreal-engine/>

## Spatial partitioning

In almost any type of simulation with agents, you will at some point need to query for other nearby agents. A good example is if you want agents to avoid each other. Using the simplest approach, you could go over all agents, and compare their distances. Doing this for one agent would be fine, but doing this for every agent will result in an algorithm of O(n²). With 10 agents, that would be 100 distance calculations, but with 20 agents, it would already be 400 calculations. This can result in huge performance issues when there are large numbers of agents in the simulation.

Luckily, there are spatial partitioning data structures that can help to optimize this process, since they organize their data based on positions, which in turn allows us to for example find the closest position, without calculating all distances. There are many different types of spatial partitioning structures, but they are usually either flat or hierarchical.

Flat structures are the easiest to understand and implement. They just divide the world in an array of cells and every position in the data structure is put in the according cell. If you then want to find other nearby positions, you just have to look through that cell and the neighboring ones. If the positions are moving, you will of course have a certain cost for updating which cell each position is in.

Hierarchical structures are more complex since they also take the density of the positions into account. There will for example be more cells in areas where there are more positions. This is usually more optimal when the positions aren’t distributed uniformly and there are lots of empty spaces. The structure is more complicated to construct and update, but allows for faster evaluation of the data.

Spatial partitioning is a requirement for any type of simulation with many agents, and therefore is an important aspect in a battle simulator game.

## Multithreading

One of the more obvious ways of optimization is utilizing multiple threads to divide the work that one thread would otherwise have to do. Usually it makes the most sense to only multithread calculations that take the most time, since multithreading can be the cause of bugs and other issues if implemented poorly.

In the case of a battle simulator, most likely it will be the navigation, avoidance and target finding that will benefit the most from being multithreaded.

### Multithreading in Unreal Engine

Multithreading in Unreal Engine is very similar to standard C++, with Unreal having their own implementations for things like parallel for and mutex. A very good explanation of the different implementations can be found on this forum post here: <https://forums.unrealengine.com/t/multithreading-and-performance-in-unreal/1216417>

The engine runs of few processes on a different thread by default, such as rendering. But most game logic such as the game tick, runs on a single thread, so there is a lot of room for optimization there.

An important thing to be aware of is that some functionality is required to run on the game thread, such as changing the location of actors.

## Rendering/animations

When working in a 3D game engine, rendering can be a potential bottleneck, especially when rendering a large amount of separate objects.

Along with rendering, animations can also take a lot of CPU frame time. Especially since skeletal animations are most commonly used, and require the CPU to evaluate the bones every frame.

The research presented here was mostly focused on how to optimize rendering and animations in Unreal Engine 5 specifically.

### Niagara

Niagara can be used to render more than just visual effects. Using smart particles, you can render particles as if they were actors, and even add basic gameplay functionality to them.

A good example of how to use Niagara to render agents is in this video from Epic Games: <https://www.youtube.com/watch?v=CqXKSyAPWZY>

It also explains other techniques of optimizing rendering large amounts of agents. Like using LODs and even billboards to render far away agents, and using vertex animations instead of skeletal ones.

### Animation Budget Allocator

Unreal Engine has a multiple systems to help optimize animations, one of which is the animation budget allocator. This system will dynamically lower tick rate of skeletal meshes that are further away from the camera, depending on the total budget it has. This means you can easily hard-limit the CPU time that is spent on updating animations. It is very easy to implement as it only involves switching skeletal meshes with budgeted skeletal meshes, and setting up the budget you want to give.  
An important negative is that if you would exceed the budget too much, you might notice lower frame rates even on meshes that are closer.

This system is perfect for when you have more than average, but still relatively small amounts of skeletal meshes.

There is an article in the Unreal Engine documentation that explains how to set this system up:  
<https://docs.unrealengine.com/5.3/en-US/animation-budget-allocator-in-unreal-engine/>

### Animation Sharing

Animation sharing is another system Unreal Engine has available to help optimize skeletal meshes.

Instead of each skeletal mesh evaluating their own animation individually, they can share it with each other. This works by evaluating one animation instance for every state required, and then each skeletal mesh component requests animation data from the animation sharing manager, which is also based on which state the mesh owner is in.

This system requires a bit of work to set up correctly, especially if you want it to seem as if each mesh has their own animation or you want to use animations that trigger on certain events, like attacking.

Unreal Engine has a documentation page dedicated to this system here:  
<https://docs.unrealengine.com/5.3/en-US/animation-sharing-plugin-in-unreal-engine/>

### Vertex Animations

Vertex animations are an alternative to skeletal animations, the biggest advantage they have is that they run almost completely on the gpu.  
Vertex animations store animation data in a texture, so the shader reads the texture to know where each vertex of the mesh needs to be at what frame in the animation.  
A big disadvantage is that you lose a lot of cool things you can do with skeletal animations, like blending. It is still possible to blend between different animations, but you need separate textures for every possible blending scenario.

There is also extra labor required to create vertex animations. You usually first create a skeletal animation, and then afterwards use a tool to bake the animation to a texture.

Unreal Engine has a plugin called AnimToTexture, that allows you to bake animation sequences to textures.

A good guide on how to use this tool is this community tutorial from Epic Games:  
<https://dev.epicgames.com/community/learning/tutorials/daE9/unreal-engine-baking-out-vertex-animation-in-editor-with-animtotexture>

## GPU programming

Whenever the CPU is your biggest bottleneck, while the GPU is still relatively free, it can be a good idea to delegate workload from the CPU to the GPU. The GPU is also able to do certain operations much more efficiently than the CPU, which can be another reason to use one of the following techniques.

However doing calculations on the GPU can add a lot of complexity to your code and should only be done if you need insane performance optimizations. For example in Ultimate Epic Battle Simulator 2, where there are millions of units in a battle, they achieved this by running most of their calculations on the GPU.

This is a very complex topic and could easily be its own research paper, if you want more sources to delve deeper into this topic, you can find references here: add link to refs!!!!!!!!!!!!

### Compute Shaders

Compute shaders are shaders that instead of rendering, are used for computations. They work very similar to normal shaders, except that their output is not used for rendering.

Because they run on the GPU, they can do some operations much more efficient than the CPU. They also run on a separate thread, so they won’t block the thread from which you call them, like with multithreading.

### CUDA

CUDA is an API developed by Nvidia, to allow for programming on the GPU. It has the same purpose as compute shaders, but instead of writing a shader, you can use a programming language like C, C++ or Fortran. This often makes it easier or more intuitive to use than compute shaders.

Since this is developed by Nvidia, it will only run on Nvidia GPUs.

You can find more information about CUDA here: <https://en.wikipedia.org/wiki/CUDA>

### Opencl

OpenCL is very similar to CUDA, but can run on both Nvidia and AMD GPUs since it is developed by [Khronos Group](https://en.wikipedia.org/wiki/Khronos_Group). It is a more general parallel programming API, but its main use is GPU programming.

You can find more information here: <https://www.khronos.org/api/opencl>

# Research

**In the research section, you detail the elements of your experiment(s), the tests, objects you will test upon and subjects you will test with, the data gathering, data cleaning or feature extraction, measurements, … and you present the results obtained in an objective manner for each of the tests you conducted.**

# case study

**Alternatively, as opposed to research, you might have opted for a case study. Whichever you choose, you detail the elements of your experiment(s), the tests, objects you will test upon and subjects you will test with, the data gathering, data cleaning or feature extraction, measurements, … and you present the results obtained in an objective manner for each of the tests you conducted.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

**Approach**

Profiling with UE insights, in dev build, cause we need it to be able to use insights, this will of course have an impact on performance

**Execution**

Simple battle Simulator

Mass battle Simulator  
Explanation of mass already done, so talk about execution, specifically what I did  
Lots of code snippets

Rendering/LODs  
Model switch  
LOD generation

Multithreading

In hindsight, it was probably not needed to multithread things like attacking

Spatial Partitioning

Animation Sharing

## introduction

**In the introduction, you write the background of your topic, explain the purpose of the paper more broadly, and explain the hypothesis, and the research question(s).**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

## Modelling

### Blockout

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.



Figure 1 : MAKING OF THE HOBBIT: THE DESOLATION OF SMAUG – LAKETOWN (WETA DIGITAL, 2014)

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

### Zbrush

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

## Texturing

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

## Shading

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

## Lighting

L Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

# Discussion

**In this section, you offer an interpretation of the results you obtained and try to relate them to the theoretical framework you presented. This is typically not a very long section, but obviously one of the most important ones.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

Talk about results, what were the biggest performance hoggers, how did we fix it?

# Conclusion

**In this section, you ascertain the demonstrable outcomes of your study and outline the merits of the project for the academic field and the discourse community. This is typically not a very long section, but obviously also one of the more important ones.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

Talk about render thread being biggest leftover issue, along with shadows

# Future work

**This section is sometimes standalone, sometimes incorporated in the conclusion. It looks at the shortcomings of the study, alternative strategies, and what could be the next course of action in the research field. This is typically not a very long section.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

Gonna be long section

Want to do:

Rendering with Niagara and vertex animations

GPU shaders

Projectiles

Deeper look into mass

# Critical Reflection

**This section is typically associated with a bachelor paper, not other forms of serious writing. It allows the student to reflect on the learning outcomes, both academically and in terms of personal growth.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas.

Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis.

Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

I expected to do more, but the time limit made it so this gradwork was a bit out of scope

I thought it would be easier to get bigger battles (10000+ agents), since I already got that far with a 2d game

I underestimated how much work the cpu needs to do to render

I learned a lot about mass, it is a very interesting system and I could probably optimize a lot more by going into the details of this system

I learned a lot about Unreal Engine source code as well, since that was my main source of information about Mass, since there are very few tutorials and documentation about more advanced Mass things.

Overall I wish I had more time, but I learned a lot already.

# References

**In this section, you list all the references you made in alphabetical order; consequently adhere to the referencing style you have chosen.**

Add all links to Zotero & export

# Acknowledgements

**In this section, you can thank people who contributed to your work in a meaningful way.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis. Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

Thank supervisor

Thank coach

Thank Howest & DAE

# Appendices

**In many cases, there are items that were developed for a research paper that can’t go into the actual paper in full. Things suc as code, art pieces, output of statistical analysis, questionnaires, … In this section, you can present these elements; use the first page to list and number the items, then paste them sequentially. If some items are too large, you can store them online, and link to them. Common practice is to keep those links active at least one year after the publication of the thesis.**

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum ac quam nec arcu semper dignissim. Nulla quam magna, varius sit amet pharetra et, dictum quis elit. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Proin ullamcorper, ipsum sit amet scelerisque rhoncus, leo quam rhoncus elit, sit amet ullamcorper tellus nisi eget sapien. Suspendisse potenti. Ut non justo viverra, tempus felis vitae, elementum mi. Morbi at dui sed lacus fringilla condimentum. Duis non odio ac arcu volutpat vehicula eu et turpis. Praesent vitae magna ante. Nulla in orci lacus. Donec quis vestibulum mi. Sed ipsum sapien, pretium maximus purus sed, bibendum consequat lectus. Aliquam porttitor dolor eu gravida vulputate. Vestibulum ut urna eget massa tincidunt ultricies. Morbi hendrerit sapien at diam tincidunt semper. Aliquam ut quam dictum quam maximus tempor sed at felis.

Add excel file

Add github link